EMC® Documentum®
Dynamic Delivery Services
Version 10.5

Installation Guide

EMC Corporation
Corporate Headquarters:
Hopkinton, MA 01748-9103
1-508-435-1000
www.EMC.com

Copyright © 2008-2014 EMC Corporation. All rights reserved.

Published February 2014

EMC believes the information in this publication is accurate as of its publication date. The information is subject to change without notice.

THE INFORMATION IN THIS PUBLICATION IS PROVIDED AS IS. EMC CORPORATION MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WITH RESPECT TO THE INFORMATION IN THIS PUBLICATION, AND SPECIFICALLY DISCLAIMS IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Use, copying, and distribution of any EMC software described in this publication requires an applicable software license.

For the most up-to-date listing of EMC product names, see EMC Corporation Trademarks on EMC.com.

Adobe and Adobe PDF Library ae trademarks or registered trademarks of Adobbe Systems Inc, in the U.S. and other countries. All other trademarks used herein are the property of their respective owners. All other trademarks used herein are the property of their respective owners.

Documentation Feedback: Your opinion matters. We want to hear from you regarding our product documentation. If you have feedback about how we can make our documentation better or easier to use, please send us your feedback directly at IIGDocumentationFeedback@emc.com.

Table of Contents

Preface 2
Purpose of the manual 2
Related documentation 2
XML Technologies Community on EDN 3
Overview 4
Introducing Dynamic Delivery Services 4
Planning and preparing 5
Preparations 5
Installing 6
Installation 6
Demo applications 9
Setting up the Eclipse IDE 14
Upgrading 16
Upgrading an existing DDS installation 16
Upgrading applications from DDS 6.5 to 6.5 SP1 16
Upgrading applications from DDS 6.5 SP1 to 6.5 SP2 18
Upgrading applications from DDS 6.5 SP2 or higher 19
Troubleshooting 20
Troubleshooting 20

Preface

Purpose of the manual

This document contains information and instructions on installing EMC Documentum Dynamic Delivery Services (DDS).

Intended audience

This guide is written for those responsible for the installation of DDS and related software.

DDS includes or uses separate products, some of which have their own installation guides. To install DDS and to develop and deploy DDS applications on a server machine, you should be familiar with the operating system(s) of the server(s) and client(s), and be able to install and configure products and services included with, or used with DDS.

Revision history

The following changes have been made to this document:

Revision	Description
July 2008	Initial publication
December 2008	Service Pack 1 update
July 2009	Service Pack 2 update
January 2010	Service Pack 3 update
September 2010	Version 10.0.0 update
May 2011	Version10.1.0 update
November 2011	Version 10.2.0 update
February 2012	Version 10.2.1 update
December 2012	Version 10.3.0 update
April 2013	Version 10.4.0 update
February 2014	Version 10.5.0 update

Related documentation

Documentation

Available documentation for DDS includes:

- Installation Guide
- Release Notes
- User Manual

- Administration Guide
- Templates Guide

Document files

Addtional information about included software, licensing, and API usage can be accessed in the `DDS/doc` folder. The folder includes the following files:

File	Description
eula.txt third-party licenses.txt	Licensing information
api/index.html	API documentation for developers.

EMC Online Support

For product documentation and support materials, including White Papers and Technical Advisories, refer to EMC Online Support (https://support.emc.com). Check regularly for new and updated documentation to ensure that you have the latest system information.

Note: Documentation installed, or packaged with the product on the download center, is current at the time of release. Documentation updates made after a release are available for download from EMC Online Support (https://support.emc.com).

XML Technologies Community on EDN

EMC Developer Community (EDN)

Software developers can find and share ideas, code, and information on the XML Technlogies community at:

https://community.emc.com/community/edn/xmltech

Overview

Introducing Dynamic Delivery Services

EMC Documentum Dynamic Delivery Services (DDS) provides a framework for developing XML delivery applications with minimal effort. DDS applications typically run in a web browser, served by a scalable delivery platform based on EMC Documentum xDB.

DDS is suited to organizations that need to deliver large and varied quantities of personalized content.

DDS can be used alongside EMC Documentum Content Server, using data feeds from its Site Caching Services (SCS). SCS is part of the EMC Documentum Site Delivery Services product set, and prefetches content and metadata to a standards-based repository.

Planning and preparing

Preparations

Before starting with DDS installation, you should have assimilated this Installation Guide and other relevant product documentation, ensure that you have the relevant software and materials to hand, that you have the requisite priviliges and resources on the server(s) and client(s) required for the installation process, and that any and all relevant backups have been performed.

Installing

Installation

Installing DDS involves several installation and configuration steps. Installation places the software on the host, and configuration prepares the system for use.

If you are already have a DDS installation on your system, you should read the Upgrading section of this document, and the Release Notes document.

DDS is shipped as a zip file (root = 'DDS'), which must be unzipped into the folder where DDS is to be installed. The following subfolders will appear:

- apache-ant
- applications
- bin
- doc
- lib
- src
- templates

1. Install and configure pre-required software.

   Note: A complete DDS installation requires additional software. The DDS hardware and sofware requirements are listed in the DDS Release Notes document. Required software includes EMC Documentum xDB and supported versions of JDK, Google Web Toolkit (GWT) and Apache Tomcat. Such additional software packages usually come with their own set of installation instructions and documentation.

   a. Install JDK, Standard Edition.

      After installation, make sure that the environment variable *JAVA_HOME* points to the location of the JDK installation.

   b. Install GWT.
   c. Install Apache Tomcat.

      This should only be done if you are planning to deploy your developed XML applications on your own system (for example for testing purposes).

      If you will be using an application server on another system, skip this step.

   d. Install xDB.
2. Unzip the compressed DDS distribution into the DDS installation target directory.
3. Check, and where necessary, change the settings in the following DDS/bin files:
   a. ant-environment.properties

      Used by the Ant targets provided with DDS, this file contains locations of DDS-related products: GWT and xDB.

b. ant-xdb.properties

Used by the Ant targets provided with DDS, this file contains settings necessary for Ant targets to connect to xDB.

An overview of the available ant tasks is included in DDS/bin/build.xml.

The properties that need to be specified in this properties file concern the xDB federation and database where applications and their data reside (See the Administration Guide for more about this). One such database can hold any number of DDS applications and their datasets (see the User Manual for information on database organization).

Note: Do NOT use the backslash character in file paths in these properties files.

4. Install and deploy the DDS Admin Tool.

DDS ships with a web application called DDS Admin Tool, for use in application administration tasks, such as application, store, and data set management. The DDS Admin Tool can manage multiple applications, stores and data sets.

The Admin Tool has to be configured to know in which database in which federation to store its own data into, and under which user account. Follow these steps to configure these matters:

a. Using the xDB Administrator Client, create a database for the DDS Admin Tool to use. For any user added (including the Administrator), be sure to remember the username and the password.

If you already have a database configured, you may skip this step.

b. At the command prompt, navigate to `DDS/bin`.
c. Enter: `dds-ant build -Dapp=admin`.

This will build a web application named admin.

During build, you will be prompted for:

- xDB bootstrap

  Enter the location of a bootstrap file of a database federation, or the URL of a running database server.

  Default is the value from `DDS/bin/ant-xdb.properties`.

- The name of the xDB database to connect to.

  Enter the name of the database that you created earlier.

  Default is the value from `DDS/bin/ant-xdb.properties`.

- The name of the xDB user that the admin application will use to connect to the database.

  Default is the value from `DDS/bin/ant-xdb.properties`.

- The password of the xDB user that will be used by the admin application to connect to the database.

  Default is the value from `DDS/bin/ant-xdb.properties`.

- The number of xDB cache pages that should be allocated for the DDS Admin Tool (see the xDB documentation for this).

  Default is the (reasonable) value from `DDS/bin/ant-xdb.properties`.

You will probably want to keep these values consistent with the ones entered in ant-xdb.properties.

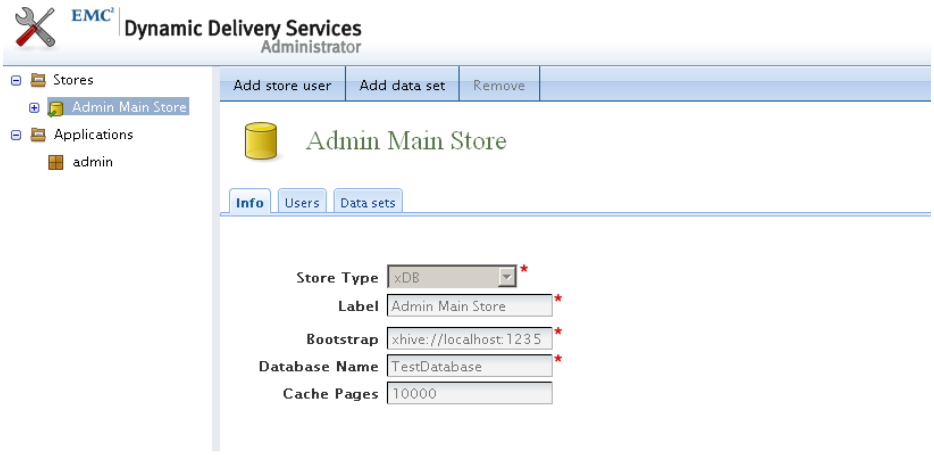
d. Enter: `dds-ant war -Dapp=admin`.

   This will create a file named `DDS/build/admin.war`.

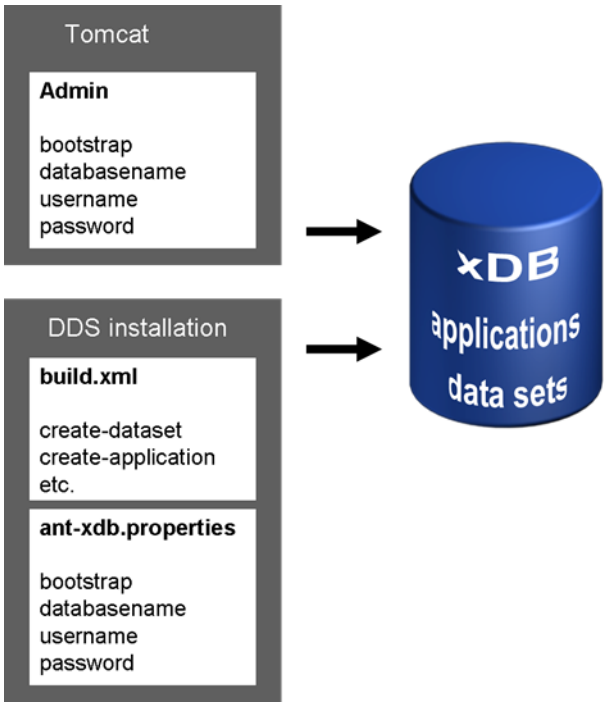
e. Deploy this war file to the Tomcat server.

   Tomcat includes a manager tool you can use for deployment.

   If Tomcat is installed locally, and not otherwise configured, the web application is now available at `http://localhost:8080/admin`, and should look something like this:



After these steps, the situation should be such that both the deployed DDS Admin Tool and the ant targets in your DDS installation can connect to the appropriate database (see figure below).



The system is now set up to start developing web-based XML applications with DDS.

Demo applications

DDS ships with a set of demo applications, with source code, to showcase DDS features and to provide developers with examples.

These can be built and deployed using Ant tasks, in the same manner as your own web applications:

1. Build the application.
2. Load the application's data into its XML Store.
3. Create a war file.
4. Deploy the war file to the target Tomcat server.

However, as you will see below, each application, its stores, and its datasets should first be configured using the DDS Admin tool, before you call these ant targets for it.

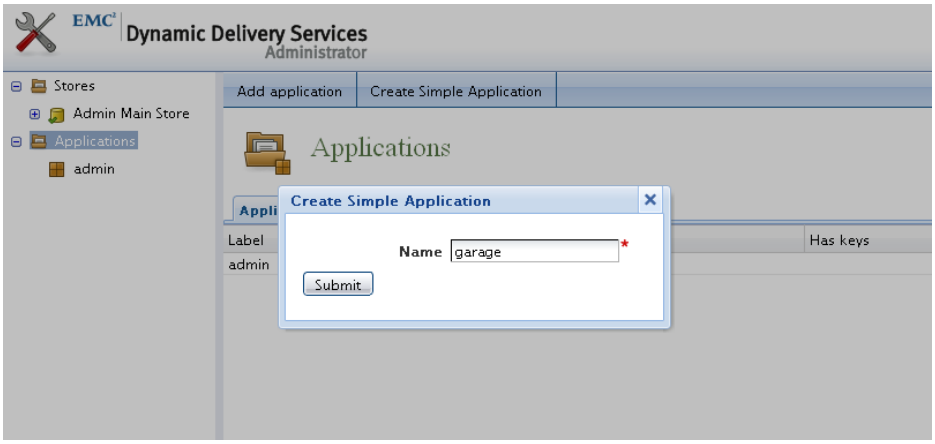
DDS demo applications include:

- garage: a basic end user web application, showing various useful programming constructs. Basic end-user documentation is provided.
- kitchensink: shows usage of DDS GUI components in a tabbed view, in the style of the original GWT Kitchen Sink demo, using data from the garage demo.
- taglib-demo: shows usage of tag libraries for data retrieval from XML Store. The sources of the JSPs are in `DDS/applications/taglib-demo/war`.
- logicengine: shows usage of the S1000D Process Data Module Logic Engine.
- webservices-demo: a demo of how to use the WSDL for XQuery and XProc functionality in DDS.

Note: When naming applications, it is recommended you use lower case characters only.

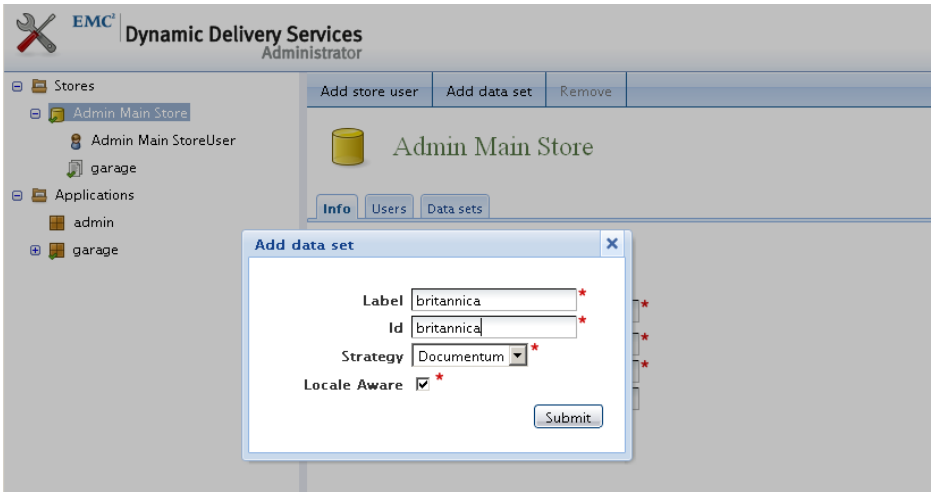
Follow the steps below to make the demo applications available to DDS developers:

1. Set the properties of the DDS installation.
2. Install and deploy the DDS Admin Tool (see previous section).
3. Use a command prompt, and navigate to `DDS/bin`.
4. To build and deploy garage, perform the following steps:
   a. In the DDS Admin Tool, go to Applications in the tree, select Create Simple Application from the menu, provide "garage" (all lowercase) as input, and click Submit.



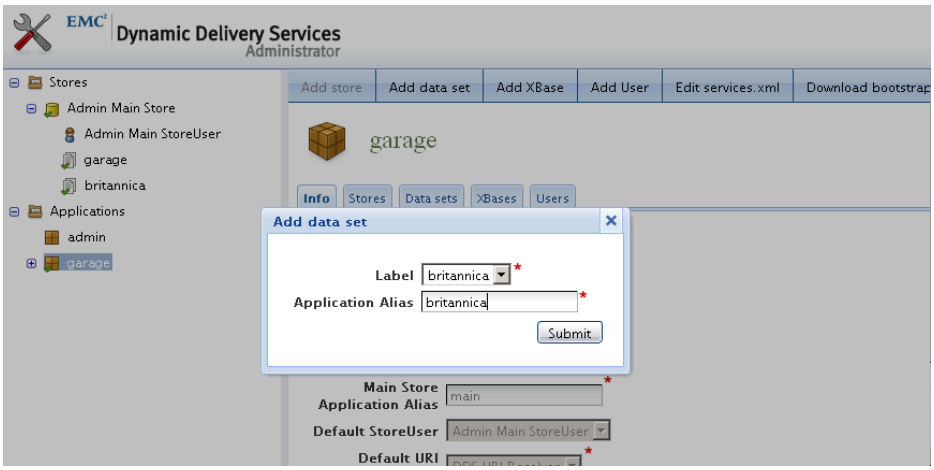
This creates an application and a locale-aware dataset both named "garage".

   b. In the DDS Admin Tool, go to Stores | Admin Main Store in the tree, and select Add data set from the menu. Enter "britannica" (all lowercase) as both Label and Id, and click Submit.



This creates a new dataset named "britannica". The garage demo uses two datasets, "garage" and "britannica".

c. In the DDS Admin Tool, go to Applications | garage in the tree, and select Add data set from the menu. Select/enter "britannica" both as Label and Application Alias, and click Submit.



This enables the garage demo to access the britannica dataset under the name "britannica".

d. From the command line: dds-ant build -Dapp=garage
e. dds-ant load-all-data -Dapp=garage
f. dds-ant hosted -Dapp=garage

This step is optional: it runs the application in hosted mode.

g. dds-ant war -Dapp=garage

This command creates the file DDS/build/garage.war.

h. Deploy the war to the Tomcat server.

If Tomcat is installed locally, and not otherwise configured, the deployed web application is now available at: `http://localhost:8080/garage`.

5. To build and deploy taglib-demo, perform the following steps:
   a. In the DDS Admin Tool, go to Applications in the tree, select Create Simple Application from the menu, provide "taglib-demo" (all lowercase) as input, and click Submit.

   This creates an application and a locale-aware dataset both named "taglib-demo".

   b. from the command line: dds-ant build -Dapp=taglib-demo
   c. dds-ant load-all-data -Dapp=taglib-demo
   d. dds-ant war -Dapp=taglib-demo

   This command creates the file DDS/build/taglib-demo.war.

   e. Deploy the war to the Tomcat server.

   If Tomcat is installed locally, and not otherwise configured, the deployed web application is now available at: http://localhost:8080/taglib-demo/index.html.

6. To build and deploy kitchensink, perform the following steps:

a. In the DDS Admin Tool, go to Applications in the tree, select Create Simple Application from the menu, provide "kitchensink" (all lowercase) as input, and click Submit.

   This creates an application and a locale-aware dataset both named "kitchensink".

b. dds-ant build -Dapp=kitchensink
c. dds-ant load-all-data -Dapp=kitchensink
d. dds-ant hosted -Dapp=kitchensink

   This step is optional: it runs the application in hosted mode.

e. dds-ant war -Dapp=kitchensink

   This command creates the file DDS/build/kitchensink.war.

f. Deploy the war to the Tomcat server.

   If Tomcat is installed locally, and not otherwise configured, the deployed web application is now available at: `http://localhost:8080/kitchensink`.

7. To build and deploy logicengine, perform the following steps:
   a. In the DDS Admin Tool, go to Applications in the tree, select Add Application from the menu, and enter "logicengine" both as Name and Label. Select "Admin Main Store" as Main Store, enter "main" as Main Store Application Alias, and select "Admin Main StoreUser" as Default StoreUser; then click Submit.

      This creates an application named "logicengine" with no datasets and no users.

   b. In the DDS Admin Tool, go to Stores | Admin Main Store in the tree, and select Add data set from the menu. Enter "datamodules" (all lowercase) as both Label and Id, unselect the Locale Aware checkbox. Click Submit.

      This creates a dataset named "datamodules" that is not locale-aware.

   c. In the DDS Admin Tool, go to Applications | logicengine in the tree, and select Add data set from the menu. Select/enter "datamodules" both as Label and Application Alias, and click Submit.
   d. In the DDS Admin Tool, go to Applications | logicengine in the tree, and select Add User from the menu. Enter "admin" as Id and "secret" as Password. Select "Admin Main StoreUser" as StoreUser and click Submit.
   e. In the DDS Admin Tool, go to Applications | logicengine in the tree, and select Edit services.xml. Uncomment the TokenService and the UserService entries, and add an entry for the LogicEngineService so that the configuration looks like in the example below, then click Submit.

```
<services>
  <service>
    <type class="DDSServiceType">USER</type>
    <name>UserService</name>

<className>com.emc.documentum.xml.dds.user.impl.UserServiceImpl</className>

  </service>
  <service>
    <dependency>UserService</dependency>
    <type class="DDSServiceType">TOKEN</type>
    <name>TokenService</name>

<className>com.emc.documentum.xml.dds.user.impl.TokenServiceImpl</className>

  </service>
  <xprocservice>
```

```
    <type class="DDSServiceType">XPROC</type>
    <name>XProcService</name>

<className>com.emc.documentum.xml.dds.xproc.impl.XProcServiceImpl</className>

  </xprocservice>
  <logicengineservice>
    <type class="DDSServiceType">LOGICENGINE</type>
    <name>LogicEngineService</name>
    <className>
      com.emc.documentum.xml.dds.le.impl.LogicEngineServiceImpl
    </className>
    <dataModuleResolver>

com.emc.documentum.xml.dds.logicengine.server.DemoDataModuleResolver

    </dataModuleResolver>
    <stateManager>

com.emc.documentum.xml.dds.logicengine.server.DemoStateManager
    </stateManager>
    <s1000DVersion>3.0</s1000DVersion>
  </logicengineservice>
</services>
```

f. In the DDS Admin Tool, go to Applications | logicengine in the tree, and select Download bootstrap XML from the menu. Save the application-bootstrap.xml file in the war/WEB-INF/classes folder of the logicengine application.
g. In the DDS Admin Tool, go to Applications | logicengine in the tree, and select Download security keys from the menu. Open the ZIP archive and store the files DDSPrivateKey.dat and DDSPublicKey.dat in the war/WEB-INF/classes folder of the logicengine application.
h. dds-ant build -Dapp=logicengine
i. dds-ant load-all-data -Dapp=logicengine
j. dds-ant hosted -Dapp=logicengine

   This step is optional: it runs the application in hosted mode.

k. dds-ant war -Dapp=logicengine

   This command creates the file DDS/build/logicengine.war.

l. Deploy the war to the Tomcat server.

   If Tomcat is installed locally, and not otherwise configured, the deployed web application is now available at: `http://localhost:8080/logicengine`.

8. To build and deploy webservices-demo, perform the following steps:
   a. In the DDS Admin Tool, go to Applications in the tree, select Create Simple Application from the menu, provide "webservices-demo" (all lowercase) as input, and click Submit.

      This creates an application and a locale-aware dataset both named "webservices-demo".

   b. dds-ant build -Dapp=webservices-demo
   c. dds-ant load-all-data -Dapp=webservices-demo
   d. dds-ant war -Dapp=webservices-demo

      This command creates the file DDS/build/webservices-demo.war.

e. Deploy the war to the Tomcat server.

   If Tomcat is installed locally, and not otherwise configured, the deployed web application is now available at: `http://localhost:8080/webservices-demo`. This URL is used by the Java client (see below). Note that using this URL in a browser has no effect: webservices-demo is not a browser application.

f. dds-ant -f ../applications/webservices-demo/bin/build.xml run -Dapp=webservices-demo

   This command runs the Java client for the web services (XQuery and XProc) to show their functionality.

9. For instructions on how to build and deploy applications from the DDS application templates, see the Templates Guide.

Setting up the Eclipse IDE

Setting up the Eclipse IDE is not required for development with DDS. Although other development tools can also be used, using the Eclipse IDE is recommended. Project files for Eclipse are included in the DDS distribution.

1. Install the Eclipse IDE.

   Eclipse IDE for Java developers can be downloaded from: http://eclipse.org/downloads/

2. Install the GWT Eclipse plugin

   The GWT Eclipse plugin can be downloaded from: .

   With the default settings, the GWT plugin will report an error for projects that use a “non-standard” location for the build output directory. As this is the case with the DDS GWT applications, you will have to disable this behavior in order to be able to use the GWT plugin to run DDS applications:

   a. Within Eclipse, open the menu **Window**, **Preferences**.
   b. Open the tree items **Google**, **Errors/Warnings**.
   c. Make sure that the setting **Incorrect build output directory** is set to either **Warning** or **Ignore**.

3. Set up the classpath variables within Eclipse.
   a. Within Eclipse, open the menu **Window**, **Preferences**.
   b. Open the tree items **Java**, **Build Path**, **Classpath Variables**.
   c. Make sure a variable `XDB_HOME` exists that points to the xDB folder.
4. Import the DDS project.
   a. Within Eclipse, open the menu **File**, **Import...** .
   b. Select **General**, **Existing Projects into Workspace**.
   c. Select the DDS installation directory as root directory.

      Refresh the projects list if the `DDS` project does not appear in the list.

   d. Click **Finish** and then **OK**.
5. Import the DDS applications.
   a. Within Eclipse, open the menu **File**, **Import...** .
   b. Select **General**, **Existing Projects into Workspace**.

c. Select the `applications` directory within the DDS installation directory as root directory.

   Refresh the projects list if the projects do not appear in the list.

d. Click **Finish** and then **OK**.

Eclipse is now set up to start developing web-based XML applications with DDS.

Upgrading

Upgrading an existing DDS installation

If you already have a DDS installation on your system, we suggest you proceed as follows:

1. Rename your current DDS folder (for example, to DDS-old).
2. Install the new DDS version in a new DDS folder.
3. Copy each of your own applications (that is, NOT the ones that came with DDS) from your DDS-old/applications folder to DDS/applications.
4. To prepare your old applications to function properly with later service packs, follow the upgrading instructions below for each one.

Related links

- Upgrading applications from DDS 6.5 to 6.5 SP1
- Upgrading applications from DDS 6.5 SP1 to 6.5 SP2
- Upgrading applications from DDS 6.5 SP2 or higher

Upgrading applications from DDS 6.5 to 6.5 SP1

DDS Service Pack 1 introduces some changes to configuration settings, aimed at making it more transparent to get and use datasets. These changes affect some class attributes, as well as some name and id elements, and introduce a new alias element.

Manual edits

As a result, the following files in previous DDS applications require `manual editing` before their re-deployment:

- application-bootstrap.xml (filesystem)
- Services.xml configuration file (applications)
- Structures.xml configuration file (applications)
- DataSet configuration files (data)

To edit the files that are stored in the applications and data sets in the database, the DDS administrator application can be used, or the admin client that is shipped with the database, whichever you prefer.

For each application, re-build, re-create the war, and re-deploy after editing its files.

Application-bootstrap.xml

Some names of child elements of the mainStore element have been changed. What was formerly called the id of a Store is now called the alias of the Store. Similarly, the name of a Store is now called id.

The `<id>[...]</id>` element must be renamed to `<alias>[...]</alias>.`

The `<name>[...]</name>` element must be renamed to `<id>[...]</id>`.

Changes of class attributes:

- `<mainStore class="com.emc.documentum.xml.dds.persistence.xdb.XDBStore">`

  to: `<mainStore class="XDBStore">`
- `<defaultStoreUser class="com.emc.documentum.xml.dds.persistence.xdb.XDBStoreUser">`:

  to: `<defaultStoreUser class="XDBStoreUser">`
- `<defaultStructureStrategy class="com.emc.documentum.xml.dds.structure.strategy.DocumentumStructureStrategy"/>`

  to: `<defaultStructureStrategy class="DocumentumStructureStrategy"/>`
- `<defaultURIResolver class="com.emc.documentum.xml.dds.uri.resolver.DDSURIResolver"/>`

  to: `<defaultURIResolver class="DDSURIResolver"/>`

Services.xml

The Services.xml configuration document for an application can be found in its Main Store, at:`/APPLICATIONS/<applicationName>/configuration/Services.xml`.

The following class attribute change must be made for every Service configured in the document:

- `<type class="com.emc.documentum.xml.dds.service.DDSServiceType">`

  to: `<type class="DDSServiceType">`

Structures.xml

The Structures.xml configuration document for an application can be found in its Main Store, at : `/APPLICATIONS/<applicationName>/configuration/Structures.xml`.

Some names of child elements of the datasetref element have been changed. What was formerly called the id of a DataSet is now called the alias of the DataSet . Similarly, the name of a DataSet is now called id.

The `<id>[...]</id>` element must be renamed to `<alias>[...]</alias>.`

The `<name>[...]</name>` element must be renamed to `<id>[...]</id>`.

The `<storeId>[...]</storeId>` element must be renamed to `<storeAlias>[...]</storeAlias>.`

DataSet Configuration Files

Each DataSet in a Store has a configuration file at:

`/DATA/<dataSetName>.xml`.

The name child element of dataset elements is now called id.

The `<name>[...]</name>` element must be renamed to `<id>[...]</id>`.

Upgrading applications from DDS 6.5 SP1 to 6.5 SP2

Changes to DDS

DDS Service Pack 2 requires some changes to the content of:

- application-bootstrap.xml
- configuration files Structures.xml and Stores.xml
- user configuration files

In addition to these changes, the following files have moved to another location within the application folder:

- application-bootstrap.xml
- DDSPrivateKey.dat and DDSPublicKey.dat
- index.html
- application css

Furthermore, DDS Service Pack 2 now generates an Eclipse configuration and a launch file for each application.

DDS offers ant target `upgrade-6.5-sp2` to help you upgrade an application from SP1 to SP2. This ant target upgrades both files on the file system and documents in the DDS main store.

Changes to DDS dependencies

DDS Service Pack 2 requires:

- xDB 9.x
- GWT 1.6.4
  Note:
  On Windows, using Java 5 to compile applications with this GWT version may result in out-of-memory errors. Java 6 is recommended.

Before you run the upgrade ant target, you must first install these new versions.

Upgrade task

To execute the upgrade task, perform the following step in the DDS/bin directory:

```
dds-ant upgrade-6.5-sp2 -Dapplication=${your application.name}
```

You will have to upgrade each application separately.

Upgrading application code

Due to the API changes related to the move to GWT 1.6 in SP2, you may need to update the application code to deal with deprecation warnings and possible compilation problems. Please follow the instructions in the DDS API documentation.

Upgrading applications from DDS 6.5 SP2 or higher

Changes to DDS

If you are coming from DDS 6.5 SP2 or any higher version, the new version requires no upgrades to configuration files on the file system or in the database. Hence, after following the steps as described in Upgrading an existing DDS installation , there is no additional upgrading to do with regard to existing applications.

Changes to DDS dependencies

Before you run any ant targets, ensure that your system meets the minimal hardware and software requirements described in the DDS Release Notes.

Troubleshooting

Troubleshooting

This section lists basic things to do, to avoid or to properly diagnose issues with DDS: the admin application, the demo applications, the Ant commands, or an application developed by yourself.

- Sources of Information

  When experiencing a problem with DDS, there are various places to look for diagnosic information.

  a. Tomcat logs

     DDS logs a lot of information. The logs folder within the Tomcat installation on your system is a valuable source of information for tracking down an issue.

  b. Tomcat console

     Depending on how logging is configured, some information may end up on the Tomcat console.

  c. Hosted mode development shell

     When running an application in hosted mode, GWT will log issues in the GWT Development Shell.

  d. Command window, console

     When running an Ant task from the command line, issues will be logged on the console.

- Configuration Settings

  In general, it is also a good idea to verify your DDS configuration settings:

  a. ant-environment.properties

     This file (in `DDS/bin`) contains the paths to software installations that DDS critically depends upon.

  b. ant-xdb.properties

     This file (in `DDS/bin`) contains the settings necessary for Ant targets to provide for proper connection to the XML store.

- Issues and Suggestions

  Below is a list of potential issues and their suggested solutions. Consult this list when encountering an issue.

a. Building an application (using dds-ant build) fails with message: “Unable to delete directory...”

   - This can be caused by orphaned instances of the Adobe Reader, left behind by the GWT hosted mode. Open the Windows Task Manager and terminate any active Adobe Reader processes.

b. Building an application (using `dds-ant build`) fails.

   - The values in `DDS/bin/ant-environment.properties` might not be set to their proper values. This causes jar-files needed for successful building not to be found.

c. Loading an application's data (using `dds-ant load-all data`) fails.

   - The values in `DDS/bin/ant-xdb.properties` might not be set to their proper values. In that case the data import module will not be able to connect to the XML Store.
   - Verify that the XML Store is actually running
   - Make sure that the database specified in `DDS/bin/ant-xdb.properties` actually exists. If not, you can create it using `dds-ant create-database`, or through the XML Store administrator.

d. Building an application (using `dds-ant build`) succeeds, but many things go wrong during running.

   - Check your application's `application-bootstrap.xml` file (in `DDS/applications/.../bin`). In particular, check the database connection configuration in there: is the store's name correct? is the default store user valid?
   - If you are running in "deployed" mode (i.e. your application is deployed under an application server), you may want to check whether the folder/file structure of your deployed application is as expected. When using Tomcat, this can be verified by looking up your application in tomcat's webapps folder, and going through the subfolders. In `WEB-INF/classes`, for example, you probably want to see `DDSPrivateKey.dat`, `DDSPublicKey.dat`, and `application-bootstrap.xml`. In `WEB-INF/lib` you'll probably expect all jars needed by your application.
   - Verify that the XML Store is actually running.

e. The garage/kitchensink/logicengine application "hangs".

   - When building the application, and loading the data, did you take care to use all lowercase characters for the application's name? (e.g. `dds-ant build -Dapplication=garage`).

f. The DDS Admin Tool seems to be "empty".

   - When building the admin application (using `dds-ant build -Dapplication=admin`), you were prompted for database connection information multiple times. On each prompt, did you enter the correct information? This can be verified by looking at the deployed application in your application server (see previous item).